IN THE SPECIFICATION:

Please amend the specification as follows.

On page 1, before the section "FIELD OF THE INVENTION", insert as a new section:

Cross-Reference to Related Application

This application claims the benefit as a continuation of U.S. Application No. 09/540,754, filed March 31, 2000 and allowed March 17, 2003, which application is incorporated herein in its entirety.

The paragraph beginning on page 1, line 6, is amended as follows:

Many computer, systems use multiple levels of caches to cache data from a memory device. For example, a computer system may have a level one cache (L1) and a larger level two cache (L2), in addition to an even larger ~~a~~ RAM memory. The L1 cache typically contains a copy of information that was previously loaded from RAM by the processor, and the L2 cache typically contains both a copy of information in the ~~Ll~~L1 cache and other information that had been loaded from RAM by the processor less recently than the information in the L1 cache.

The paragraph beginning on page 7, line 19, is amended as follows:

FIG. 4 is a flow chart that shows one embodiment of a method of retrieving data from a multilevel cache system according to the present invention. According to this method, a request for information is issued (401) to a first data array, a second data array, and a merged tag array at substantially the same time. For example, the request might be issued by CPU 101 of FIG. 1 to first data array 104, second data array 105, and merged tag array 103. The request may be generated by an instruction, such as a load instruction. Typically, the request might specify an address in a memory (such as a system RAM) where the information is stored. Information stored in a location in the first data array corresponding to the request is received (402), and one or more instructions that ~~consume~~ use the information received are processed tentatively (403). For example, the request may be for information in a certain memory address, and the first data array will have a location (i.e., a line) corresponding to that memory location. Assuming for example that the instruction was LOAD a register with the data from a certain memory location (e.g., LD B [100]), then the first data array would supply the information stored in the data array set that corresponds to the memory location (e.g., memory location 100).

The paragraph beginning on page 8, line 11, is amended as follows:

The instructions may be allowed to ~~consume~~ use the data unconditionally but, because the data is tentatively processed, the system state (or architectural state) is not yet modified. That is, the instruction is not retired. If the instruction were to LOAD register B from a memory location 100, as in the example above, then the instruction and the information retrieved from the location in the first data array may be stored in a buffer. Later, instructions in the stream may then be tentatively processed. As discussed above, a location in the first data array corresponds to the memory location from which the information was requested. However, the first data array location may correspond to multiple memory locations. Thus, the information received from the first data array may, or may not, correspond to the information in the requested memory location. If the first data array location does contain a copy of the information in the requested memory location, a cache hit occurs in the first data array. If the first data array location does not contain a copy of the information in the requested memory location, a cache miss occurs. Because the validity of the data retrieved from the first data array has not yet been confirmed, the processor executes the instructions in a tentative mode. Based on the request received, the merged tag array determines whether the request was a cache hit in the first data array and whether the request was a cache hit in the second data array (404).

The paragraph beginning on page 12, line 7 is amended as follows:

According to a further embodiment of the current invention, when a request for information is issued to a first data array, a second data array, and a tag array at substantially the same time (e.g., 401 of FIG. 4), the request may also ~~issued~~issue to a translation lookaside buffer (TLB) at substantially the same time (405). The processor assumes that the TLB has the desired information and ~~consumes~~ uses the data returned from the first level tag array. If it is later determined by the merged TLB that the request was not authorized (406), then the processor state control component may transfer control to an exception handler (407). In an embodiment, ~~If~~if it is ~~later~~ determined by the TLB that a valid translation is not present, then the processor state control component flushes the improper instructions, as discussed above, and returns the processor to the architectural state it was in prior to issuance of the improper instruction. The processor state control component does not retire instructions until the TLB determines that they are valid.

The paragraph beginning on page 13, line 11 is amended as follows:

Embodiments of the present invention provide a multilevel cache system that has multiple data arrays and a single merged tag array. Such a cache system allows cache accesses in the aggregate to be achieved at an improved speed, thus improving overall system performance, and reduces the size of the cache system. A first level data array is allowed to complete the look-up prior to determining whether there is a cache hit and the data retrieved is ~~consumed~~ used on a speculative basis. If it is later determined that the request to the first level data array was a cache miss, the instructions that were tentatively performed are flushed and loaded into the first level cache from a second level cache, or the data request may be forwarded to another level of the memory hierarchy.